2008-08-18

Autonomous Trajectory Planning Using Real-Time Information Updates

Hurni, Michael A.

The American Institute of Aeronautics and Astronautics (AIAA)

http://hdl.handle.net/10945/29644

AIAA Guidance, Navigation and Control Conference and Exhibit
18 - 21 August 2008, Honolulu, Hawaii

AIAA 2008-6305

Autonomous Trajectory Planning Using Real-Time Information Updates

Michael A. Hurni[1], Pooya Sekhavat[2], and I. Michael Ross[3]
Naval Postgraduate School, Monterey, California, 93943

We present a dynamic optimal control method for autonomous trajectory planning and control of an Unmanned Ground Vehicle (UGV) using real-time information updates. The objective of the UGV is to traverse from an initial start point and reach its goal in minimum time, with maximum robustness, while avoiding both static and dynamic obstacles. This is achieved by deriving the control solution that carries out the initial planning problem while minimizing a cost and satisfying constraints based on the initial global knowledge of the area. To combat the problem of inaccurate global knowledge and a dynamic environment, the UGV uses its sensors to map the locally detected change in the environment and continuously updates its global map to re-compute a control solution that can achieve an optimal trajectory to the goal. Simulation results illustrate successful implementation of the method in various scenarios.

I. Introduction

Autonomous trajectory planning of unmanned vehicles has been one of the main goals in robotics for several years. In recent years, this problem has become particularly important as a result of rapid growth in its applications to both military and civilian missions. Various control methods have been proposed and examined for autonomous guidance and control of unmanned vehicles.[1,2]

Sometimes it is important to distinguish between *path planning*, *motion planning* and *trajectory planning*. *Path planning* finds a feasible path from start to goal. When that computed path is parameterized by time it becomes *motion planning*. When the control solution (actual commands to the system actuators) for the computed path is determined along with the time parameterization, it becomes *trajectory planning*.[1] There are two approaches to trajectory planning for a dynamic system: The decoupled approach and the direct approach. The decoupled approach involves first searching for a path (using a path planner) and then finding a time-optimal time scaling for the path subject to the actuator limits. The direct approach searches for the trajectory directly within the system's state space.[2]

In the decoupled trajectory planning methods, path planning is the basic problem at hand. Ref. 1 defines *path planning* as identifying a trajectory that will cause a robot to reach its goal location when executed. Notice that in the above definition, the words "path planning" and "trajectory" appear in the same sentence. The words "path planning" and "trajectory planning" become blurred in the literature because of the fact that most of the more popular methods use path planners to conduct trajectory planning by either using the decoupled approach or by adapting what is traditionally a path planner to search in the vehicle's state space to perform trajectory planning. For these reasons path planning and trajectory planning are often used synonymously in the literature and in this work as well.

For the purposes of this work, we require the trajectory planner to meet the following performance specifications. It should not only solve for the path to the target, but also provide the controls necessary to reach the goal. The path to the goal should also satisfy some measure of optimality. It must be able to handle the avoidance of static and dynamic obstacles. It must be able to take into account the physical limitations of the vehicle (vehicle dynamics and operational constraints) when planning the trajectory. It should be information centric; that is, able to plan its path from start to goal with whatever global knowledge of the environment is available, and then update this global map with local snapshots obtained by the vehicle sensors during the course of the maneuver. Its

[1] Ph.D. Student, Department of Mechanical and Astronautical Engineering, Email: mahurni@nps.edu.
[2] Research Scientist, Dept. of Mech. and Astronautical Eng., AIAA Member, Email: psekhava@nps.edu.
[3] Professor, Dept. of Mech. and Astronautical Eng., AIAA Associate Fellow, Email: imross@nps.edu.

computational performance should be such that it can be implemented in real-time while the vehicle is in motion to the target. Finally, the method should be complete; in other words it should guarantee a solution (if one exists).

Choosing the best trajectory planner to accomplish a given task depends on the performance parameters and criteria that are required by the designer. References 1 and 2 give detailed descriptions and analyses of the various path planning methods. These methods include Road Maps, Cell Decompositions, Artificial Potential Fields, Optimal Control and Obstacle Avoidance Algorithms. Obstacle Avoidance Algorithms include Bug Algorithms, Vector Field Histograms, the Bubble Band Technique, Curvature Velocity Techniques, The Schlegel Approach, The ASL Approach and many more. Each path planner has its advantages and disadvantages. When assessing a planner as unable to meet a specific performance specification, it is meant that the planner cannot achieve such performance on its own. For example, when stating that the control problem for the planned trajectory is not solved, it means that the planner itself does not provide the controls necessary to navigate the vehicle along the trajectory, but with the help of a trajectory following subsystem it may perform adequately. The desire is for the planner to meet the performance requirements without the help of other systems or algorithms.

Bug algorithms[1-6] represent an example of obstacle avoidance algorithms. They do not provide the control necessary to traverse the planned trajectory; in most cases they do not provide an optimal solution and they often fail in dynamic obstacle avoidance. Bug algorithms do not account for vehicle dynamics or constraints and operate based on local information only. Artificial Potential Field[1, 2, 7-20] methods also do not provide the control necessary to traverse the planned trajectory. There is nothing within the construct of the method to guarantee a measure of optimality, and vehicle constraints are not accounted for without the addition of other techniques. Roadmaps,[1, 2, 21-36] like the other methods mentioned thus far, do not directly calculate the control for the planned trajectory. They are computationally costly, and thus are not suitable for real-time applications. Probabilistic Roadmaps can be implemented in real-time, but sacrifice optimality for the faster run times. Cell Decomposition[1, 2, 37-46] methods also do not meet all the above-listed performance criteria. They do not solve the control solution for the planned path. Their computational complexity is too high when using a sufficient resolution to obtain optimality. This limitation on their computational efficiency makes dynamic obstacle avoidance difficult, and limits their ability to be updated frequently enough to be able to be used in real-time and receive and incorporate local information updates. And finally, these methods need additional techniques to account for kinodynamic and nonholonomic constraints. The Optimal Control[1, 2, 47-55] technique, on the other hand, meets all the desired performance specifications stated above that other methods fall short of, and is the technique of choice in this work. The most distinguished difference is that the Optimal Control method does not need help from other techniques/systems to achieve the desired performance. This is important since hybrid techniques often suffer from the shortcomings of all the methods used to make up the overall planner.

Optimal Control Trajectory Planning using Numerical Optimization, as described in Ref. 2, is a direct approach to the complete motion-planning problem, which determines the path to the target by searching within the vehicle's state space. The result is the complete state space and control solution from start to goal. The basic concept of how optimal path planning works follows from Refs. 2, 47. First the planner must be given the kinodynamic equations of the vehicle, the obstacles to be coded into smooth path constraint functions, and the cost function. The kinodynamic equations can also be viewed as constraints (like the obstacles), defining the relationship between the vehicle state and the control input. The obstacles need not be smooth, but the constraints used to define the obstacle must be made up of one or more smooth functions. The Optimal Control technique finds a solution to the state equations that takes the vehicle from the initial state at time zero to the final state at the final time, while avoiding obstacles, obeying vehicle state and control limits, and minimizing some cost function. The cost function can be any function of state variables, control variables and time, as long as it is sufficiently smooth (i.e. continuous and differentiable). Reference 2 states that there are two drawbacks to the Optimal Control technique. First, numerical methods require an initial bias to steer the solution in the right direction. Second, this method can be computationally costly and thus require excessively long computation times, at least from the standpoint of running the algorithm in real-time. Advances in optimal control and numerical optimization tools over the last five to ten years have overcome these drawbacks. We first address the bias that has traditionally been required in solving an optimal control problem numerically. The issue here is the fact that the need to provide a bias can take away from the autonomy of the problem, however this does not have to be the case. The initial offline run to determine a candidate trajectory does not have to be autonomous. This first run is performed prior to sending the vehicle out on its mission. Future real-time runs must be autonomous, and by virtue of already having a candidate solution from the offline run, the vehicle can use that solution as the bias to steer its first real-time solution. Subsequently, each real-time solution can be used as the bias for the next real-time run, thus preserving its autonomy. This new approach involves using the optimal control algorithm in a feedback form, thus self-generating the needed bias. References 48-55 solve various path planning problems using Optimal Control with numerical optimization in a feedback control algorithm. In Ref.

49 a novel time-optimal sampled-data feedback control algorithm is introduced for closed-loop control of NPSAT1 in the presence of disturbances. The feedback law is not analytically explicit; rather, closed-loop control is obtained by a rapid re-computation of the open-loop time-optimal control at each update instant. The basic idea of this algorithm is to take the initial conditions and desired final states (as the initial bias), and conduct an offline run; the result is an initial control trajectory and an estimate of the final time. These are then used as the bias for the real-time closed-loop runs. After the initial input of start and goal information, the system is autonomous. In Ref. 50 an optimal nonlinear feedback guidance law was constructed for a reusable launch vehicle (RLV) based on the concept of using the offline open-loop solution to bias the first real-time closed loop solution, and each subsequent closed-loop run was biased by the previous solution. Again, autonomy was preserved. In Refs. 48, 51, 52, 53, 54, 55 the use of optimal control in a feedback form using a bias that is obtained autonomously is demonstrated on an RLV, a tricycle, a UGV, a UAV, and a slew problem for NPSAT1. In reference to the computational cost, it has been shown in Refs. 48-55 that the use of a bias to help steer the solution trajectory speeds up the run times significantly from seconds to fractions of a second. So the bias does not just aid in achieving feasible solutions, but also speeds up the solution process considerably. Computation time can be further reduced by at least a factor of 100 by optimizing the actual code and eliminating the Windows and MATLAB overhead.[50] Advancements in sparse linear algebra, development of new algorithms, and improved computer processor speeds have made solving optimization problems relatively easy and fast.[51] Recent applications of real-time optimal control[48-55] have proven to be very promising in facilitating feedback solutions to complex nonlinear systems.[51]

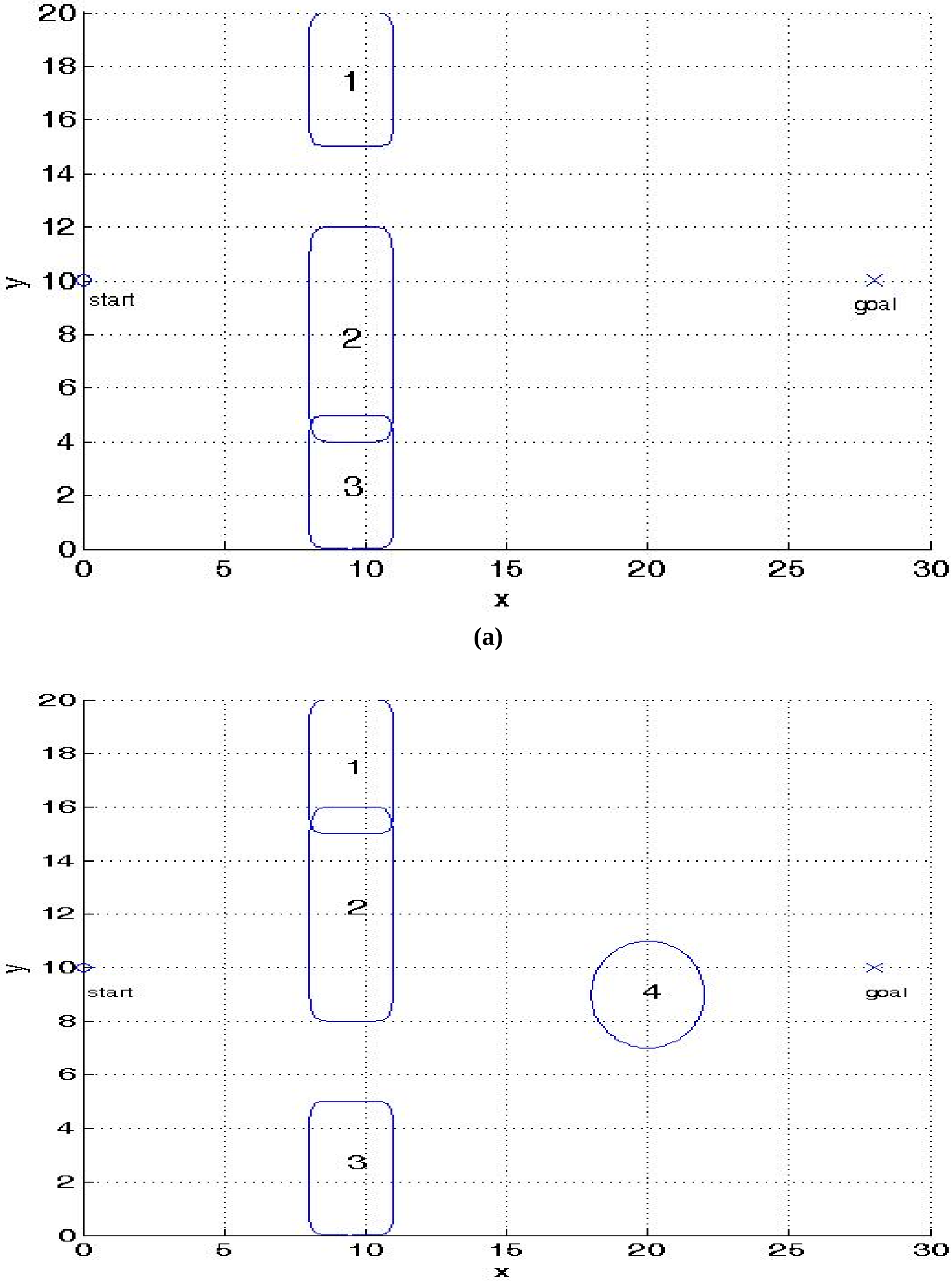
In recognizing the preceding advances, this paper presents an optimal control method for autonomous path planning and control of an Unmanned Ground Vehicle (UGV) using real-time information updates. First we present the problem, model the UGV dynamics and describe the assumptions made in the problem formulation. Next we cover the path planning method utilized, followed by the simulation results and analysis. Finally, the conclusions are presented.

II. Problem Description and Vehicle Dynamics

Figure 1(a) shows the initial configuration of the problem with all dimensional units in meters (*m*). This is what the vehicle knows *a priori* and is used to solve the initial planning problem. The UGV starts from $(x,y) = (0,10)$ and is tasked to proceed to the target point $(x,y) = (28,10)$ in minimum time, with maximum robustness, while avoiding moving and stationary obstacles. Obstacles 1 and 3 are stationary. At $t = 1$ *sec* obstacle 2 commences to move upward at a rate of *1 m/s* and continues to move until $t = 5$ *sec*. This results in the north passage being blocked by obstacle 2 while opening up the south passage. Due to obstacle overlap, the north passage will be completely blocked at $t = 4$ *sec*. At $t = 15$ *sec*, obstacle 4 appears ("pops up") as a new obstacle not known *a priori*. Figure 1(b) shows the final configuration of the obstacle environment. The instantaneous position and the size and shape of the obstacles are the only information known by the vehicle. Thus, starting with the environment shown in Fig. 1(a), environment information is updated by the vehicle each instant up to the point of Fig. 1(b).

The p-norm was used to algebraically model the shapes of the obstacles used in this work. Using the p-norm, one can easily model any square, rectangle, circle, or ellipse. These shapes are all that is needed in path planning since any obstacle can be modeled by fitting one of those shapes around it. Modeling an obstacle's exact shape and size is more complex and highly unnecessary. Equation (1) shows the general form of the equation used to model the obstacles. In Fig. 1, obstacles 1-3 were modeled as rectangles with p = 8 and obstacle 4 was modeled as a circle (p = 2).

$$h_i(x(t), y(t)) = \left| \left(\frac{x(t) - x_c}{a} \right)^p \right| + \left| \left(\frac{y(t) - y_c}{b} \right)^p \right| - \left| c^p \right| = 0 \tag{1}$$



(b)

Figure 1. Initial (a) and final (b) obstacle configurations.

The UGV that is used in this paper is modeled as a four-wheeled car with front-wheel steering.[48] Figure 2 shows the model configuration.[48] Only the front wheels are capable of turning and the back wheels must roll without slipping. $'L'$ is the length of the car between the front and rear axles. $'r_t'$ is the instantaneous turning radius.

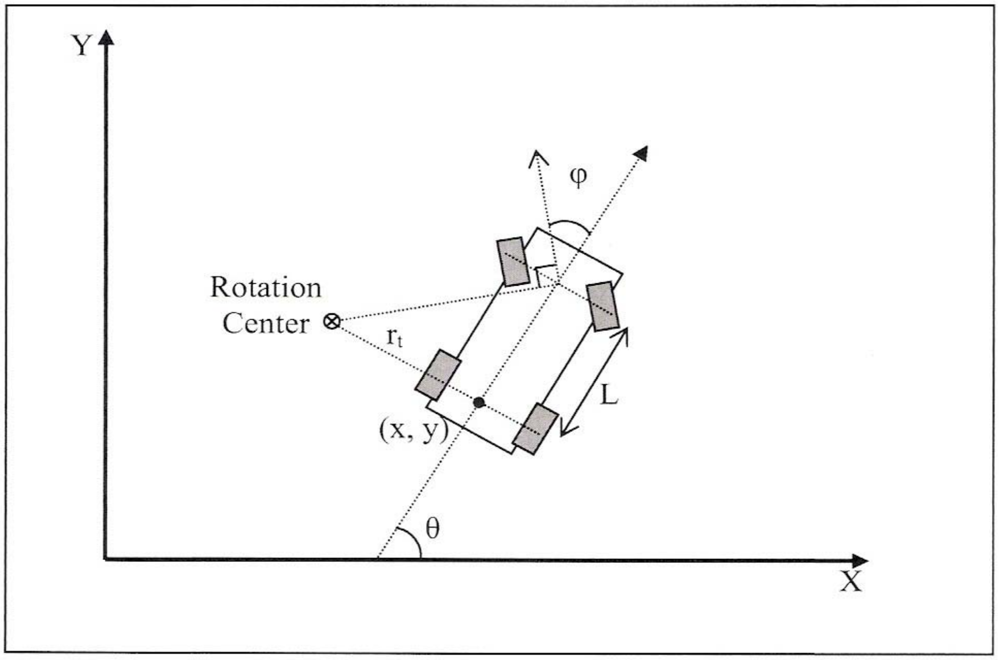


Figure 2. A four-wheeled car model with front-wheel steering.

The state vector is composed of two position variables (*x, y (m)*), an orientation variable (*θ (rads)*), the car's velocity (*v (m/s)*) and the angle of the front wheels (*φ (rads)*) with respect to the car's heading (Eq. (2)). The x-y position of the car is measured at the center point of the rear axle.

$$\underline{x} \in X := \begin{Bmatrix} x : 0 \le x(t) \le 30 \\ y : 0 \le y(t) \le 20 \\ \theta : -2\pi \le \theta(t) \le 2\pi \\ v : -1 \le v(t) \le 1 \\ \varphi : -1 \le \varphi(t) \le 1 \end{Bmatrix} \tag{2}$$

The control vector consists of the vehicle's acceleration (*a* (m/s^2)) and the rate of change of the front wheel angle (*ω (rad/s)*) (Eq. (3)).

$$\underline{u} \in U := \begin{Bmatrix} a : -0.5 \le a(t) \le 0.5 \\ \omega : -0.33 \le \omega(t) \le 0.33 \end{Bmatrix} \tag{3}$$

Using the fact that $L = r_t \tan(\varphi)$ and $v = r_t\dot{\theta}$, assigning the length $L = 0.5m$ results in the kinematic equations of motion for the UGV (Eq. (4)).

$$\dot{\underline{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} v\cos(\theta) \\ v\sin(\theta) \\ 2v\tan(\varphi) \\ a \\ \omega \end{bmatrix} \tag{4}$$

III. Trajectory Planning

The UGV must travel from the start to the target point in minimum time, with maximum robustness, while avoiding both static and dynamic obstacles and not violating its physical limitations. This problem can be posed as a dynamic optimal control problem; that is, an optimal control problem wherein the problem formulation itself changes as time moves forward. In the problem posed in this paper, the problem formulation changes as a result of information, or the lack of it, about the obstacles. To appreciate this point, recall that Eq. (1) defines an obstacle perimeter. It follows that $h_i(x(t), y(t)) < 0$ corresponds to the space inside the obstacle and $h_i(x(t), y(t)) > 0$ indicates the area outside the obstacle and therefore is the desired constraint. Setting $c = 1$ in Eq. (1) and moving it to the right hand side results in Eq. (5). The desired obstacle constraint then becomes $h_i(x(t), y(t)) > 1$.

$$h_i(x(t), y(t)) = \left| \left(\frac{x(t) - x_c}{a} \right)^p \right| + \left| \left(\frac{y(t) - y_c}{b} \right)^p \right| = 1 \tag{5}$$

Taking the natural log of both sides of Eq. (5) results in Eq. (6) and restores the obstacle constraint equation to $h_i(x(t), y(t)) > 0$.

$$h_i(x(t), y(t)) = \ln \left(\left| \left(\frac{x(t) - x_c}{a} \right)^p \right| + \left| \left(\frac{y(t) - y_c}{b} \right)^p \right| \right) = 0 \tag{6}$$

Extending this concept to all of the obstacles in the problem at hand, Eq. (7) shows the path constraint equations (with p = 8 for obstacles 1 - 3 and p = 2 for obstacle 4) for the obstacles. The path constraints for obstacles 2 and 4 are time dependent, allowing obstacle 2 to move and obstacle 4 to appear later; however, note that this information is not known *a priori*, rather, the optimal control problem is dynamically updated with this information.

$$\begin{aligned}
h_1(x(t), y(t)) &= \ln\left(\left(\frac{x - 9.5}{1.5} \right)^8 + \left(\frac{y - 17.5}{2.5} \right)^8 \right) > 0 \\
h_2(x(t), y(t)) &= \begin{cases} \ln\left(\left(\frac{x - 9.5}{1.5} \right)^8 + \left(\frac{y - 8}{4} \right)^8 \right) > 0; t < 1 \\ \ln\left(\left(\frac{x - 9.5}{1.5} \right)^8 + \left(\frac{y - 8 - (t - 1)}{4} \right)^8 \right) > 0; 1 \le t \le 5 \\ \ln\left(\left(\frac{x - 9.5}{1.5} \right)^8 + \left(\frac{y - 12}{4} \right)^8 \right) > 0; t > 5 \end{cases} \\
h_3(x(t), y(t)) &= \ln\left(\left(\frac{x - 9.5}{1.5} \right)^8 + \left(\frac{y - 2.5}{2.5} \right)^8 \right) > 0 \\
h_4(x(t), y(t)) &= \ln\left(\left(\frac{x - 20}{2} \right)^2 + \left(\frac{y - 9}{2} \right)^2 \right) > 0; t \ge 15
\end{aligned} \tag{7}$$

With regards to the maneuver performance, we want to minimize time while maximizing robustness. By robustness we mean that the vehicle must be able to execute the control solution and achieve the mission goals despite some uncertainty in the information and models without hitting any obstacles. Uncertainty in the information can come from numerous sources: Errors in sensor data, sensor accuracy, external unforeseen forces, and the error that is inherent in propagating the previous solution while calculating an updated solution. In order to minimize such uncertainties, it is necessary to derive a Robustness Function (*r(t)*) whose integral over time will be minimum when robustness is maximized. To derive *r(t)* we take Eq. (1) (which has a value of zero on the obstacle and increases as the vehicle traverses away from the obstacle) and apply a double exponential operation. This is done for each obstacle and the result is summed and then multiplied by a weighting factor (*w*). The resulting Robustness Function is as follows with *n* equal to the number of obstacles:

$$r(t) = w\sum_{i=1}^{n}\left(e^{e^{-h_i(x(t),y(t))}} - 1\right) \tag{8}$$

In the case of a single obstacle with $w = 1/5$, *r(t)* will equal 2.83 at its center, 0.344 on its edge, and continue to decrease exponentially to zero as the distance to the obstacle is increased. Thus, the cost function can be completely expressed in terms of the endpoint cost (final time) and the running cost (*r(t)*):

$$J[\underline{x}(\cdot),\underline{u}(\cdot),t_f] = t_f + \int_0^{t_f} r(t)dt \tag{9}$$

At each instant of time, the optimal control problem can be formulated as follows:

<u>Given</u>: The state and control vectors of Eqs. (2) and (3).

<u>Minimize</u>: The cost function of Eq. (9).

<u>Subject to</u>: The dynamics of Eq. (4), and:

$$\begin{aligned}
&\underline{x}(t_0) = [0,10,0,0,0]^T \\
&t_0 = 0 \\
&e(\underline{x}_f, t_f) = \begin{bmatrix} x(t_f)-28 \\ y(t_f)-10 \\ v(t_f) \\ \varphi(t_f) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \\
&-\pi/2 \le \theta(t_f) \le \pi/2 \\
&h_i(x(t), y(t)) > 0
\end{aligned} \tag{10}$$

The vehicle's path planning algorithm (Fig. 3) follows the following logic:

1. The vehicle is given an instantaneous snapshot of the environment that is known prior to the start of its mission. This environment map will act as the global map that will be used to plan the trajectory to the target. It is a dynamic map that will track environmental changes.
2. The initial path to the target (and the controls throughout that path) is solved *offline*, prior to the vehicle commencing the maneuver. This initialization calculation can take longer than future calculations in real-time, because the initialization contains only two points to aid in its solution, the initial conditions and the target conditions. This initial solution serves as the bias used to steer subsequent real-time calculations, which results in lowering calculation times.
3. The following two steps are conducted simultaneously in real-time until the vehicle reaches the target point:
    a. The vehicle sensors take a snapshot of the local environment. This snapshot (along with global information received from other sources via its communications link) is used to update the global map and the current vehicle state. The new environment map and vehicle state are used to calculate the next optimal trajectory.

This calculation runs much quicker, because the data from the previous solution is used as a bias to steer the next solution.

b. The vehicle travels toward the target using the controls from the previous solution. This is done in simulations through the application of control trajectory interpolation and state propagation using a Runge-Kutta algorithm.

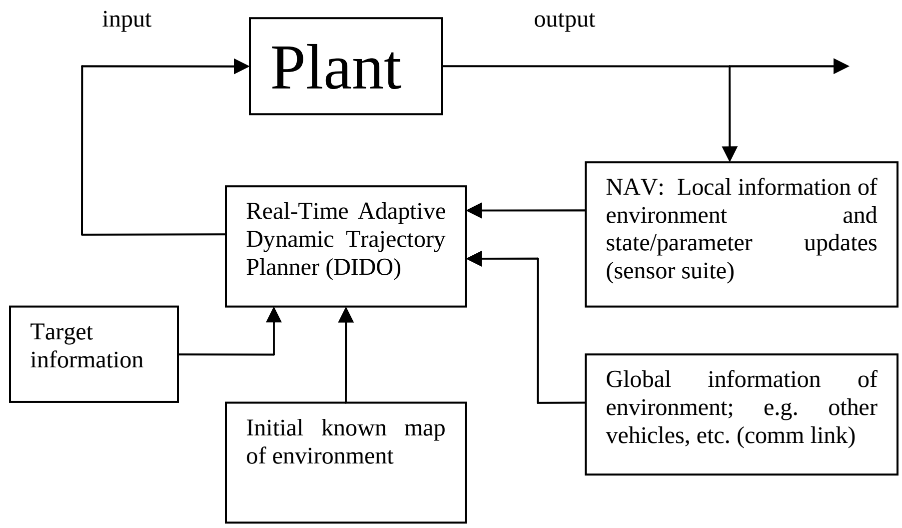


Figure 3. Path Planning Algorithm Logic.

It is clear that our planning algorithm is based on the "premise" that the optimal control problem can be solved rather rapidly. That this is indeed true is well-known in the aerospace community – thanks to rapid advances in the pseudospectral (PS) methods. The optimal control problem is solved using DIDO©, a MATLAB-based software tool that incorporates PS methods.[56] Proper formulation is the key to solving any optimal control problem. The variables must be well scaled and balanced, the dynamics and constraints must be clearly defined, and a good initial bias (although not necessary) can help speed up convergence to the solution.

IV. RESULTS AND ANALYSIS

Analyses were performed in steps to build up from the simple to the more complex scenario described in the problem formulation.

A. Initialization

Using the initial environment map of Fig. 1(a), an open-loop optimal trajectory is shown as a solid line in Fig. 4(a). This solution was obtained knowing only the start and end conditions using only minimum time as the cost. The robustness factor is excluded from the cost in this initialization run, because the vehicle is not yet moving, therefore there is no propagation error. The resulting solution will be the ideal minimum-time optimal solution.

One key aspect of PS methods that is utilized in trajectory planning is its high convergence rate. That is, in a PS method only a few discrete points, or nodes, are needed to reconstruct accurate continuous-time solutions. For the problem at hand, a mere 15-point solution is quite adequate to meet all the performance specifications. This feature is demonstrated in Fig. 4(a) where a 15-node solution (points marked with circles) is superimposed with a solution obtained with 60 nodes. That the 15-node solution is almost on top of the 60 node solution indicates that the PS solution has practically converged with 15 nodes. Hence, the 15 node solution is used as the bias to steer subsequent real-time (closed loop) calculations. The overall optimal maneuver time is *30.4 sec* and the solution satisfies all necessary conditions for optimality. The necessary conditions are not developed in this paper for the purposes of brevity; however, for the purposes of illustration we show in Fig. 4(b) a plot of the Hamiltonian function. The constancy of the Hamiltonian at the value -1 indicates that the resulting trajectory is indeed a Pontryagin extremal.

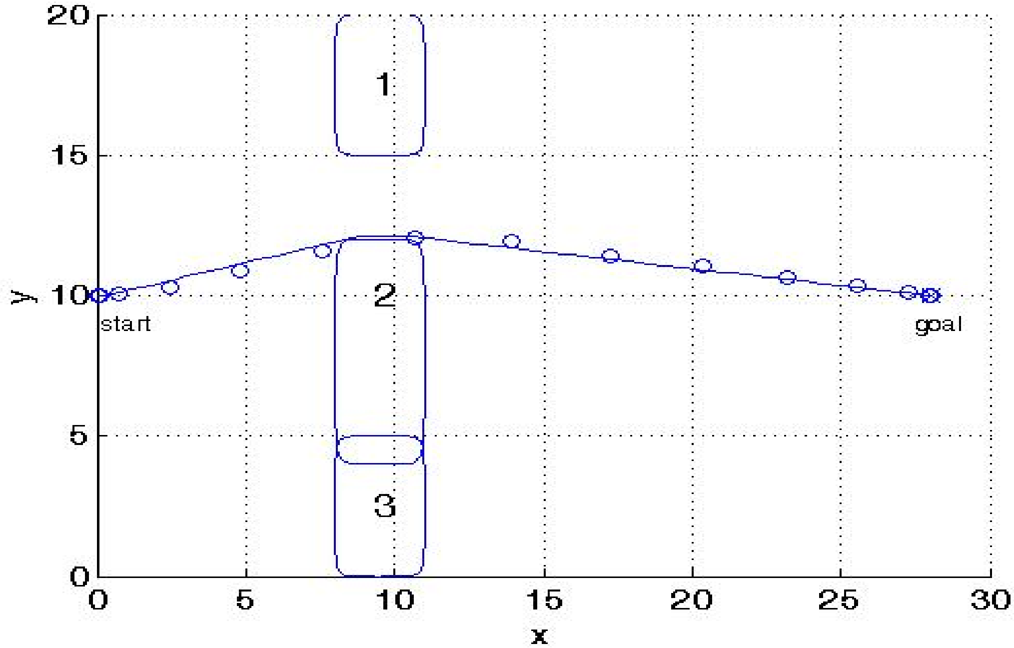


Figure 4(a). Initial 60 and 15 node *offline* runs.

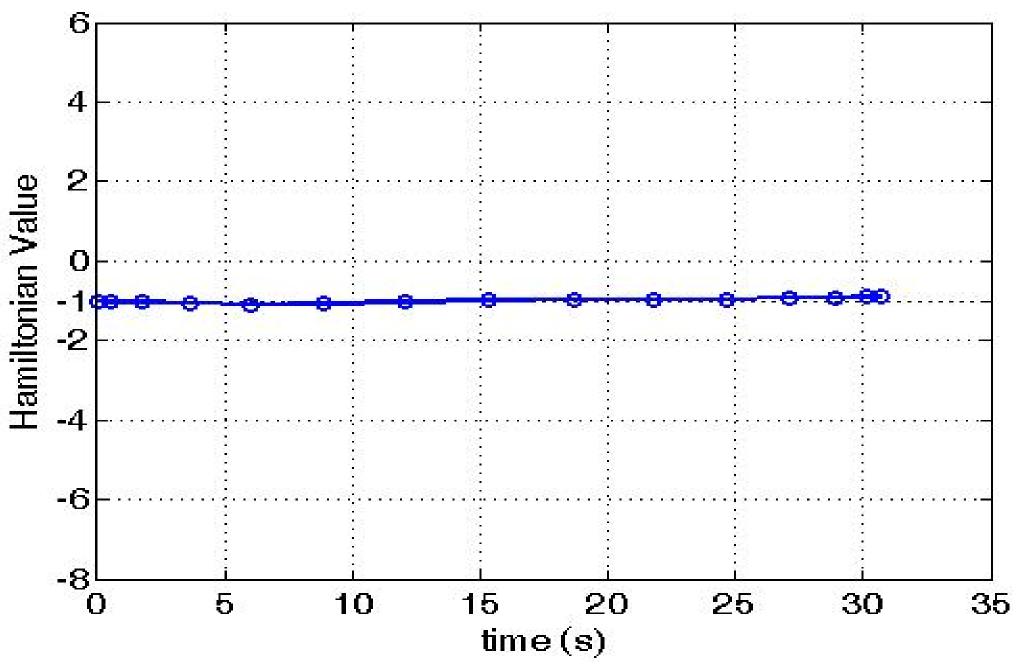


Figure 4(b). Hamiltonian Function for the Initialization Run.

B. Static Obstacles

Consider the simple case of static obstacles (see Fig. 4(a)). An execution of the path planning algorithm (see Fig. 3) results in the solution shown in Fig. 5. The overall maneuver time is *30.8 sec*. This is *0.4 sec* higher than the initialization run. The increase in maneuver time can be attributed to the fact that the robustness factor is included in the overall cost when operating in real-time. Therefore, this is no longer a minimum time problem. As evident from Fig. 5, the vehicle takes a wider turn around obstacle 2 in order to optimize the total cost. Maneuver time (endpoint cost) has risen by 0.4, but at the same time the running cost is minimized in order to maximize robustness.

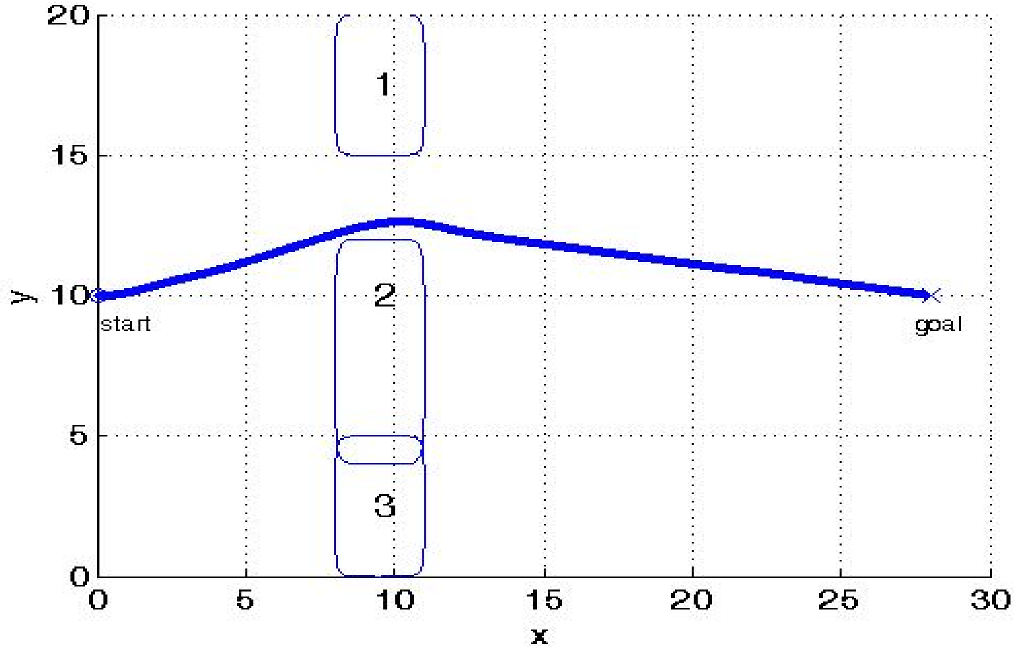


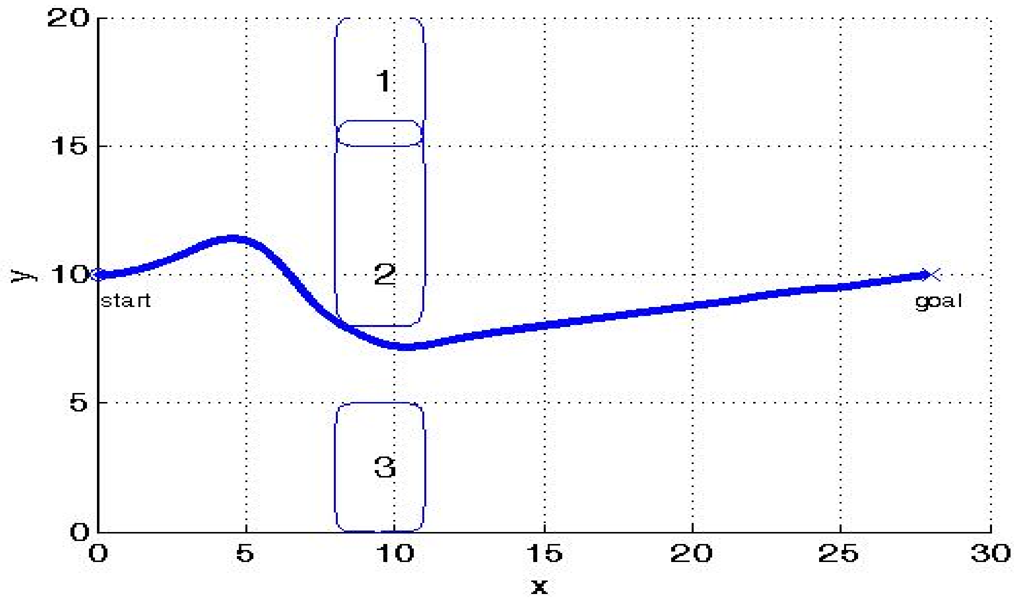
Figure 5. UGV trajectory with three static obstacles.

C. Moving Obstacles

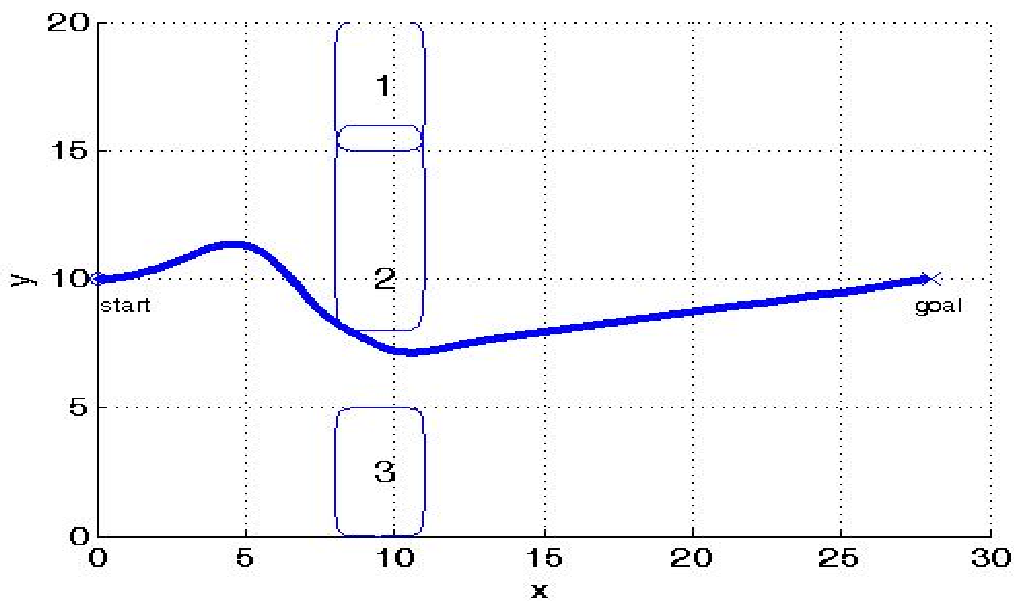
In this scenario, obstacle 2 moves up throughout the vehicle's motion. The results from the path planning algorithm are shown in Fig. 6(a). The overall maneuver time in this case is increased to *31.8 sec*. It is important to note that no prediction is involved in determining the future position of the obstacle and the vehicle changes its course autonomously. Each new run uses the snapshot of the environment taken just prior to the run. So during the time it takes to complete the new solution, although the obstacle is moving continuously, the vehicle maneuvers based on the previous solution with no knowledge of obstacle motion until the next snapshot is taken and the vehicle sees the obstacle has changed position. This can be seen by noting the concavity of the vehicle trajectory in Fig. 6(a) during the first four seconds. It shows the vehicle trying to head further north to follow the top of obstacle 2 as it moves north. One might be tempted to predict the position of the obstacle, which in this scenario would have resulted in a much earlier turn to the south by the vehicle. However, since we have no way of knowing how far the obstacle will move, trying to predict its behavior is a risky undertaking that will not necessarily be helpful.

Further study was conducted in relating the allowed calculation time per run to the robustness factor in the cost function. A maximum allowed calculation time of *0.4 sec* was used for each closed loop run. If larger run times were allowed, it would lead to larger propagation error, which could result in infeasible solutions. With larger error, a real vehicle might fail to reach its goal due to collision with an obstacle. All simulations to this point used a weighting of 1/5 for the running cost. Figure 6(b) shows the results of using a smaller weighting on the running cost (1/8 instead of 1/5). The close-up of Fig. 6(c) shows that the vehicle trajectory collides with obstacle 2. Using the 1/8 weighting that caused collision, the allowed run time was reduced to *0.3 sec* instead of *0.4 sec*. Figure 6(d) and the close-up of Fig. 6(e) show the vehicle successfully maneuvering around obstacle 2. It can be further stipulated that in the limit as run time approaches zero, the running cost can go to zero, thus making this purely a minimum time problem.

It was also necessary to include a new decision path in the algorithm that prevented the use of a northerly bias (that of the previous run) once the passage was blocked by obstacle 2. Once this condition is sensed, the vehicle must stop, the algorithm must only use the current position and goal as the bias and use a higher number of nodes (25 in this case) to generate the solution (ignoring the northerly bias). Closed loop operation will then continue again using 15 node runs. It was interesting to find out that if the path must be drastically modified (i.e. a northerly passage vs a southerly passage of obstacle 2), then using the previous solution as a bias steers the algorithm to an infeasible solution. Figure 7(a) shows a snapshot of the maneuver at t = *5.2 seconds*. This time instant was picked because obstacle 2 has completed its northward motion. Notice that the vehicle is still going north even though the north passage was completely blocked as of $t = 4$ *seconds*. The extra lag time it takes for the vehicle to turn south is due to two factors. First, the vehicle continues to move forward using the last solution as it is computing the new solution and determining the infeasibility of continuing on the northern route; and second, the vehicle constraints incorporated into the optimal control problem formulation prevent it from simply making a sharp turn to the south. The front wheels can not exceed their maximum turn rate, and the vehicle has a turning radius that must be obeyed. The initial 25 node southerly solution (Fig. 7(b)) shows the path that must be taken given the vehicle constraints and the initial northerly heading. It corresponds to the optimal path after $t = 4$ *seconds*, when the northern path becomes completely blocked.

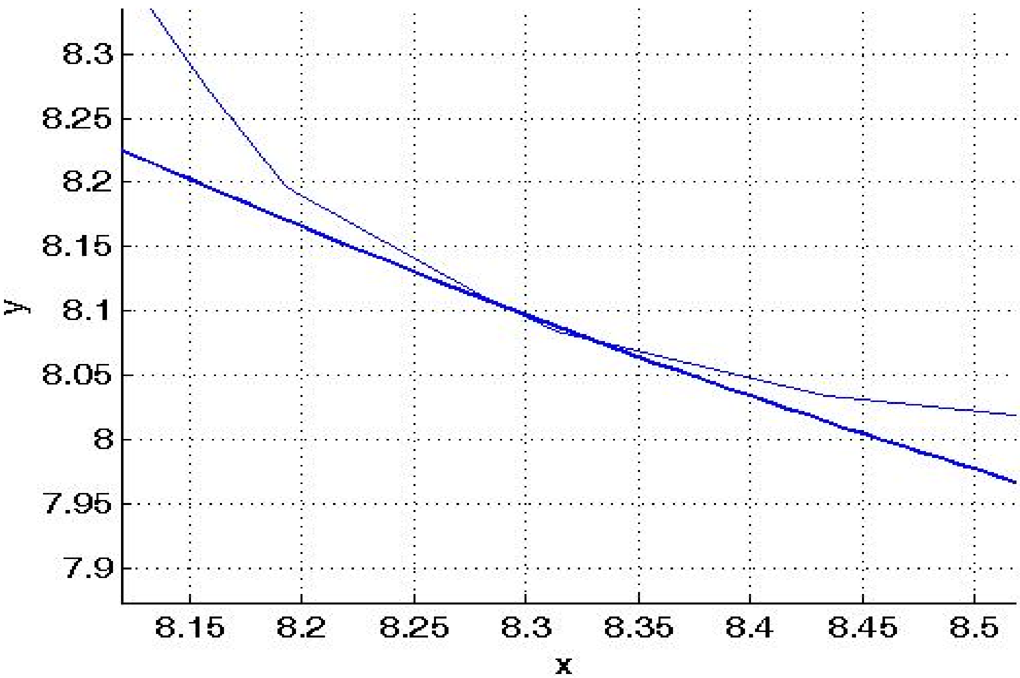


(a)

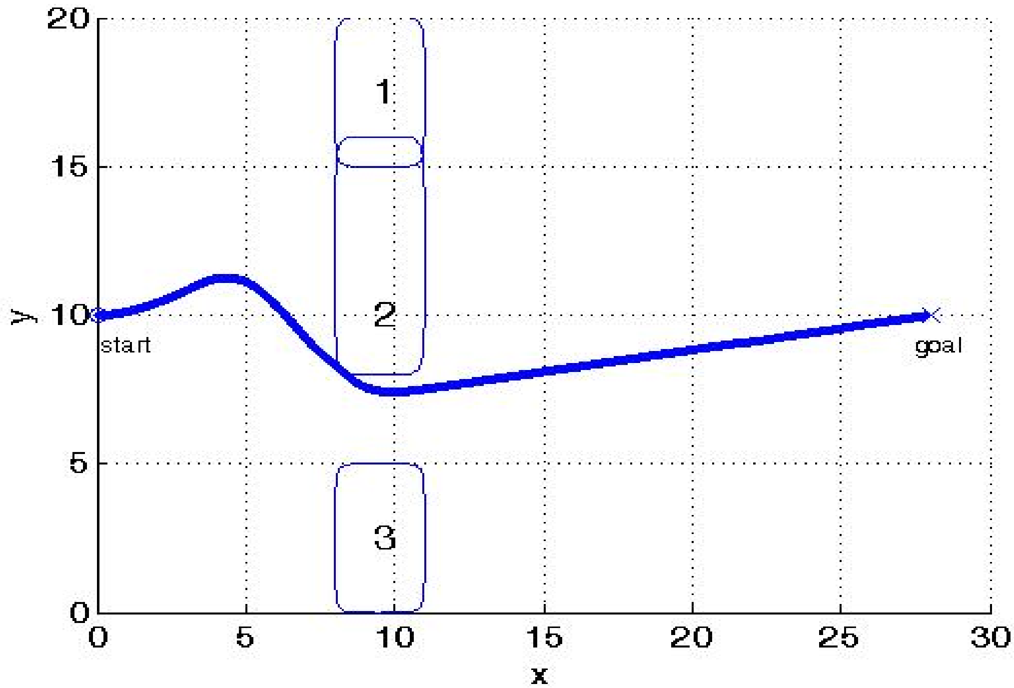


(b)

Figure 6. UGV trajectory with moving obstacles.



(c)



(d)

Figure 6. UGV trajectory with moving obstacles.

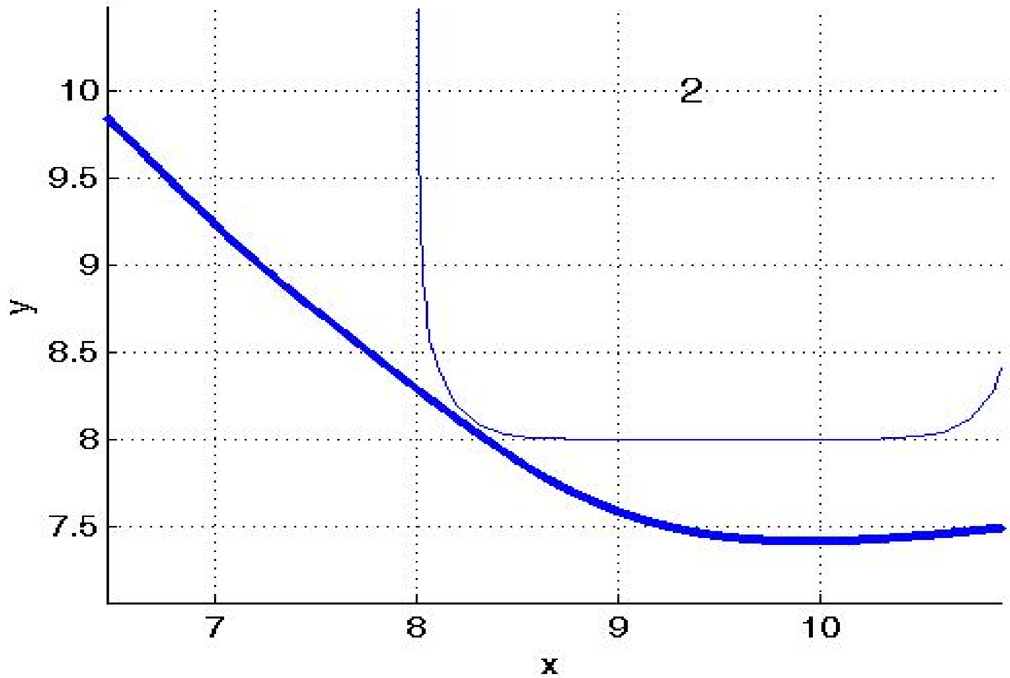


Figure 6(e). UGV trajectory with moving obstacles.

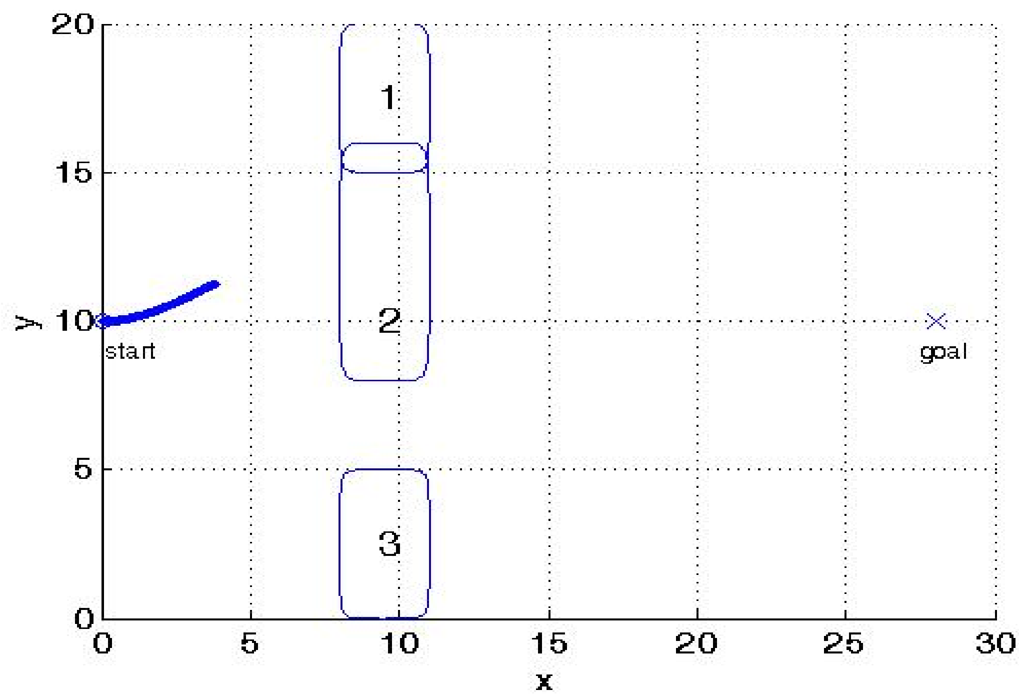


Figure 7(a). Formulation of southerly solution once north passage is shut.

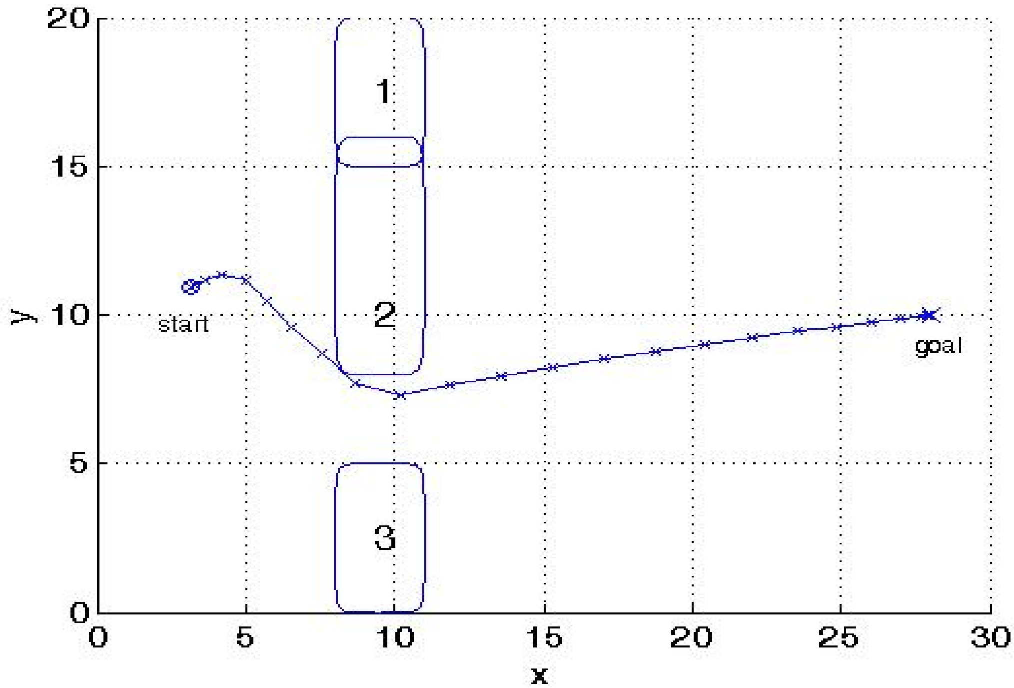
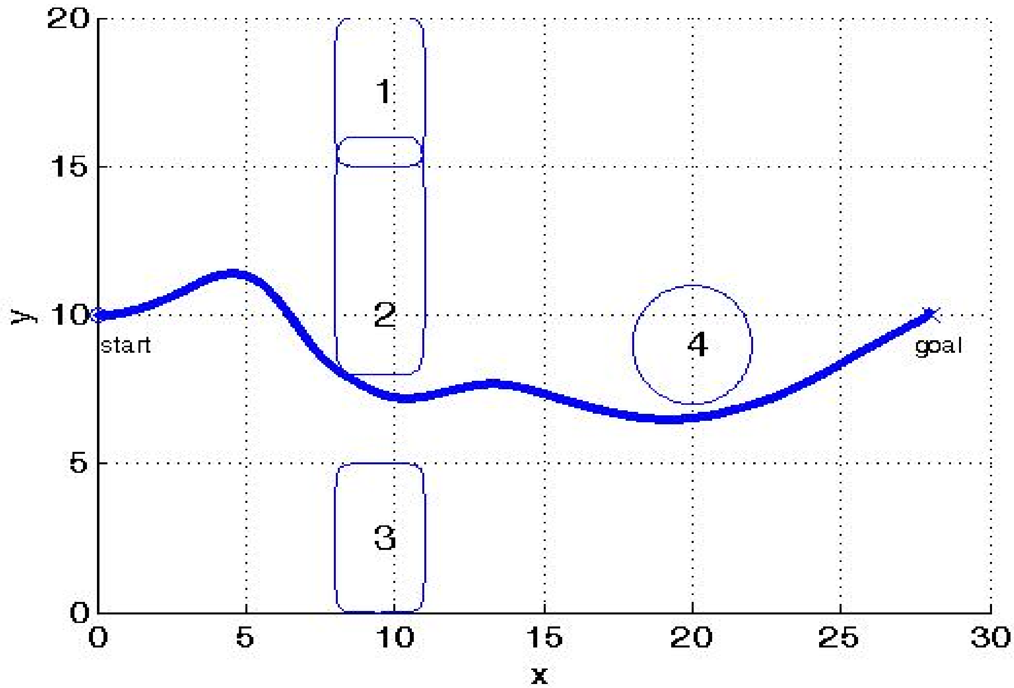


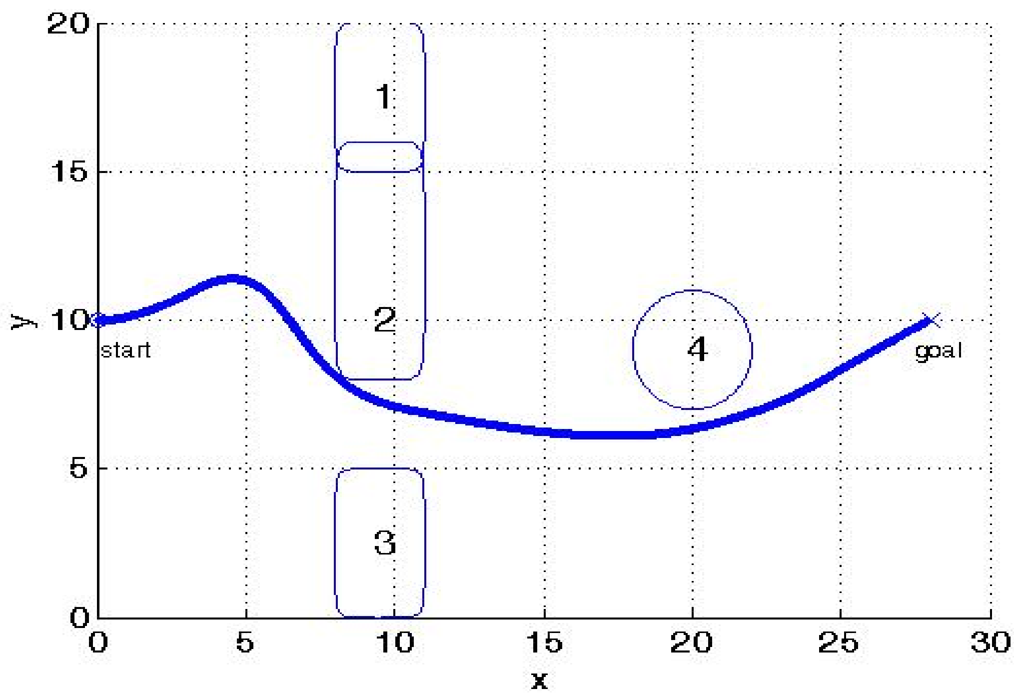
Figure 7(b). Formulation of southerly solution once north passage is shut.

D. Pop-Up Obstacles

The last scenario considered in this study involves repeating the previous scenario with the addition of a pop up obstacle at t = *15 sec*. The results of using the real-time optimal control technique are shown in Fig. 8(a). The weighting on the running cost was back to 1/5 and run times were again limited to *0.4 sec*. The overall maneuver time in this case was increased to *32.4 sec*. Initially the vehicle heads toward the opening on the north end of obstacle 2. As obstacle 2 closes the north passage, the vehicle tries to follow it over the top until the passage is completely blocked off, at which point the vehicle autonomously finds a new optimal path to the south. Once the vehicle has passed obstacle 2 it heads straight at its goal, because it does not yet know of the existence of obstacle 4. Once its sensors pick up obstacle 4, the vehicle computes a new path around that obstacle without any operator interaction or involvement. Figure 8(b) shows what the path would look like if obstacle 4 had appeared sooner (t = *9 sec*). This new scenario has an overall maneuver time of *32.3 sec*. Combining Fig. 8(a) and 8(b) into Fig. 8(c) shows that the vehicle switches to the path to avoid obstacle 4 once the existence of obstacle 4 is determined. Note that once the vehicle comes around obstacle 2, the optimal path to the goal depends on how soon the planner has knowledge of obstacle 4. The path goes directly at the goal if the existence of obstacle 4 is unknown (Fig. 6(a)). If the planner knows about obstacle 4 prior to navigating around obstacle 2, it follows the wide route around obstacle 4. Figure 8(d) combines Fig. 8(c) with Fig. 6(a) to show the vehicle switching paths from the path generated without obstacle 4 to that generated with obstacle 4.

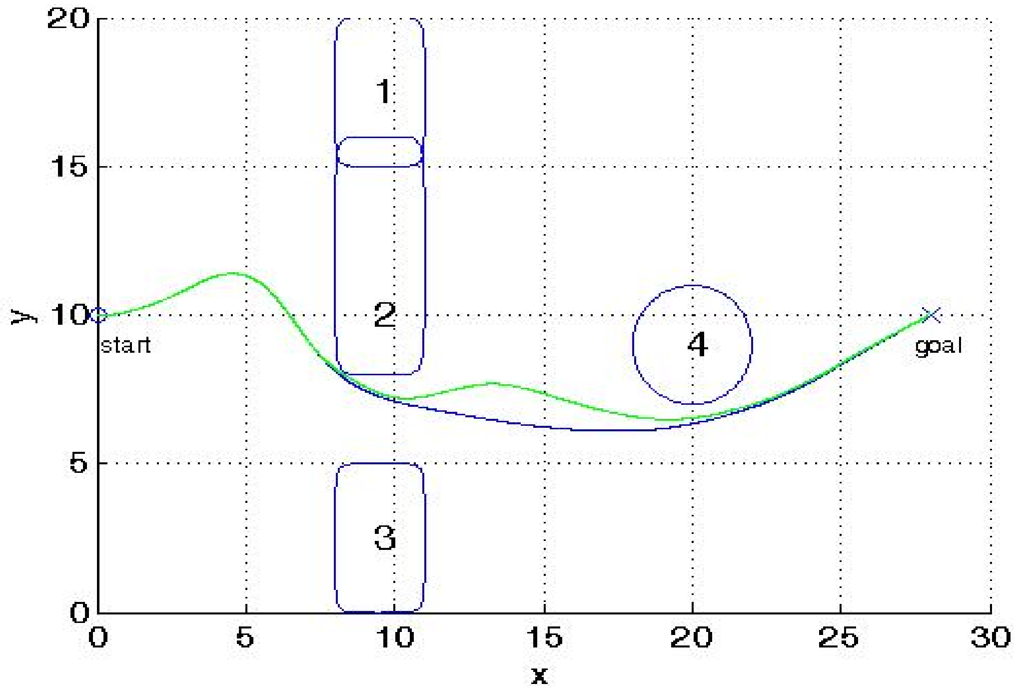


(a)

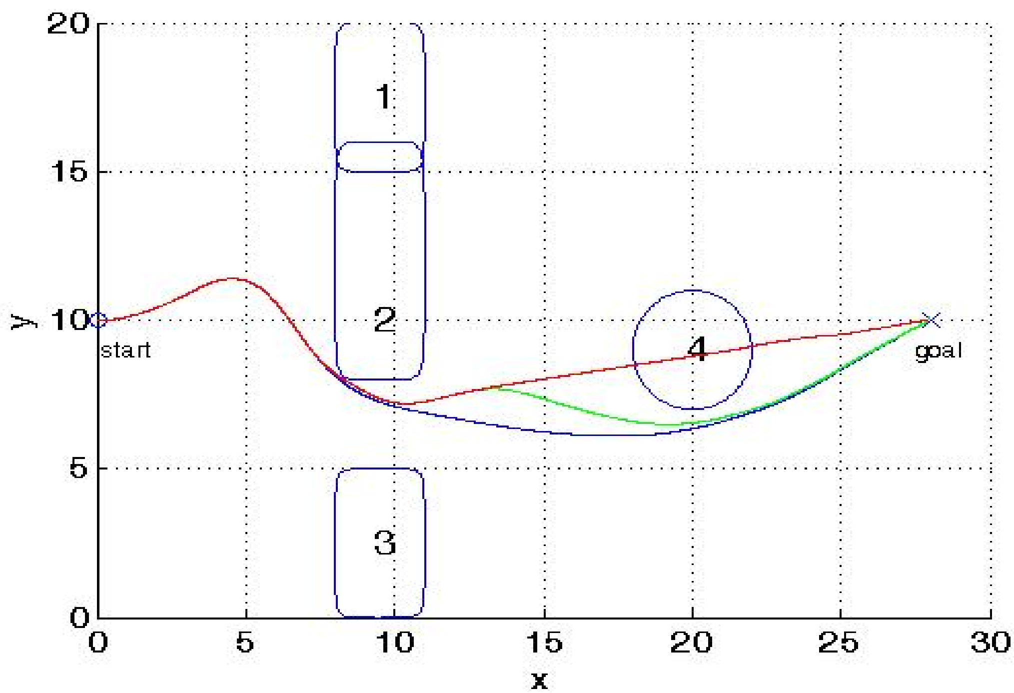


(b)

Figure 8. Otimal path in the presence of moving and pop up obstacles.



(c)



(d)

Figure 8. Otimal path in the presence of moving and pop up obstacles.

V. CONCLUSIONS

This paper presents the use of an optimal control technique for the autonomous trajectory planning of an unmanned ground vehicle while incorporating real-time information updates. It shows the utility of using an information centric planner that would continuously and autonomously update its global map of the environment using local sensory data and comm link data to achieve optimal performance.

It is shown that, given a reasonable starting bias, the pseudospectral optimal control embodied in the software package DIDO[56] can solve the optimal path planning problem in real-time. It is also shown that there is a trade-off between vehicle/obstacle dynamical limits, solution computation times and the weighting of the running cost used to provide system robustness. Furthermore, an infeasible DIDO[56] solution indicates the optimal path has changed too drastically and the previous solution should not be used as a bias for the next solution. This check for infeasibility improves autonomy by providing a new decision path based on the result of this feasibility check.

REFERENCES

[1]Siegwart, R. and Nourbakhsh, I.R., *Introduction to Autonomous Mobile Robots*, The MIT Press, 2004.

[2]Choset, H., and others, *Principles of Robot Motion; Theory, Algorithms, and Implementation*, The MIT Press, 2005.

[3]Haro, F. and Torres, M., "A Comparison of Path Planning Algorithms for Omni-Directional Robots in Dynamic Environments," paper presented at the IEEE 3rd Latin American Robotics Symposium, Santiago, Chile, October 2006.

[4]Kaman, I., Riman, E., and Rivlin, E., "Range-Sensor Based Navigation in Three Dimensions," *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, Detroit, Michigan, 10-15 May 1999.

[5]Langar, R.A., Coelho, L.S., and Oliveira, G.H., "K-Bug, A New Bug Approach for Mobile Robot's Path Planning," paper presented at the 16th IEEE International Conference on Control Applications, Singapore, 1-3 October 2007.

[6]Magid, E. and Rivlin, E., "CautiousBug: A Competitive Algorithm for Sensory-Based Robot Navigation," *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, September 28 - October 2, 2004.

[7]Vadakkepat, P., Tan, K.C., and Ming-Liang, W., "Evolutionary Artificial Potential Fields and Their Application in Real Time Robot Path Planning," *Proceedings of the 2000 Congress on Evolutionary Computation*, Vol. 1, pp. 256-263, 16-19 July 2000.

[8]Kim, D.H., Lee, H.W., Shin, S., and Suzuki, T., "Local Path Planning Based on New Repulsive Potential Functions with Angle Distributions," *Proceedings of the Third International Conference on Information Technology and Applications*, 4-7 July 2005.

[9]Zhu, Q., Yan, Y., and Xing, Z., "Robot Path Planning Based on Artificial Potential Field Approach with Simulated Annealing," *Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications*, Vol. 2, October 2006.

[10]Cen, Y., Wang, L., and Zhang, H., "Real-time Obstacle Avoidance Strategy for Mobile Robot Based on Improved Coordinating Potential Field with Genetic Algorithm," 16th *IEEE International Conference on Control Applications*, Singapore, 1-3 October 2007.

[11]Zelek, J.S. and Levine, M.D., "Local-Global Concurrent Path Planning and Execution," *IEEE Transactions on Systems, Man, and Cybernetics*, Part A, Vol. 30, No. 6, pp. 865-870, November 2000.

[12]Dinham, M. and Fang, G., "Time Optimal Path Planning for Mobile Robots in Dynamic Environments," *Proceedings of the 2007 IEEE International Conference on Mechatronics and Automation*, Harbin, China, 5-8 August 2007.

[13]Urakubo, T., Okuma, K., and Tado, Y., "Feedback Control of a Two Wheeled Mobile Robot with Obstacle Avoidance Using Potential Functions," *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, September 28 - October 2, 2004.

[14]Warren, C.W., "Multiple Robot Path Coordination Using Artificial Potential Fields," *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, Vol. 1, pp. 500-505, 13-18 May 1990.

[15]Baras, J.S., Tan, X., and Hovareshti, P., "Decentralized Control of Autonomous Vehicles," *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, Hawaii, 9-12 December 2003.

[16]Zheng, T. and Zhao, X., "A Novel Approach for Multiple Mobile Robot Path Planning in Dynamic Unknown Environment," *IEEE Conference on Robotics, Automation and Mechatronics*, December 2006.

[17]Balch, T. and Hybinette, M., "Social Potentials for Scalable Multi-Robot Formations," *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, San Francisco, CA, 24-28 April 2000.

[18]Kowalczyk, W. and Kozlowski, K., "Artificial Potential Based Control for a Large Scale Formation of Mobile Robots," *Proceedings of the 4th International Workshop on Robot Motion and Control*, pp. 285-291, 17-20 June 2004.

[19]Saaj, C.M., Lappas, V., and Gazi, V., "Spacecraft Swarm Navigation and Control Using Artificial Potential Field and Sliding Mode Control," *IEEE International Conference on Industrial Technology*, pp. 2646-2651, 15-17 December 2006.

[20]Jia, Q. and Li, G., "Formation Control and Obstacle Avoidance Algorithm of Multiple Autonomous Underwater Vehicles (AUVs) Based on Potential Function and Behavior Rules," *Proceedings of the IEEE International Conference on Automation and Logistics*, Jinan, China, 18-21 August 2007.

[21]Amato, N.M. and Dale, L.K., "Probabilistic Roadmap Methods are Embarrassingly Parallel," *Proceedings of the 1999 IEEE International Conference on Robotics & Automation*, Detroit, Michigan, May 1999.

[22]Kindel, R., Hsu, D., Latombe, J.C., and Rock, S., "Kinodynamic Motion Planning Amidst Moving Obstacles," *Proceedings of the 2000 IEEE International Conference on Robotics & Automation*, San Francisco, CA, April 2000.

[23]Sanchez, G. and Latombe, J.C., "Using a PRM Planner to Compare Centralized and Decoupled Planning for Multi-Robot Systems," *Proceedings of the 2002 IEEE International conference on Robotics & Automation*, Washington, DC, May 2002.

[24]Li, T.Y. and Shie, Y.C., "An Incremental Learning Approach to Motion Planning with Roadmap Management," *Proceedings of the 2002 IEEE International conference on Robotics & Automation*, Washington, DC, May 2002.

[25]Martin, S.R., Wright, S.E., and Sheppard, J.W., "Offline and Online Evolutionary Bi-Directional RRT Algorithms for Efficient Re-Planning in Dynamic Environments," *Proceedings of the 3rd Annual IEEE Conference on Automation Science and Engineering*, Scottsdale, AZ, 22-25 September 2007.

[26]Song, G., Miller, S., and Amato, N.M., "Customizing PRM Roadmaps at Query Time," *Proceedings of the 2001 IEEE International Conference on Robotics & Automation*, Seoul, Korea, 21-26 May 2001.

[27]Kazemi, M. and Mehrandezh, M., "Robotic Navigation Using Harmonic Function-based Probabilistic Roadmaps," *Proceedings of the 2004 IEEE International Conference on Robotics & Automation*, New Orleans, LA, April 2004.

[28]Kazemi, M., Mehrandezh, M., and Gupta, K., "An Incremental Harmonic Function-based Probabilistic Roadmap Approach to Robot Path Planning," *Proceedings of the 2005 IEEE International Conference on Robotics & Automation*, Barcelona, Spain, April 2005.

[29]Yang, Y. and Brock, O., "Adapting the Sampling Distribution in PRM Planners Based on an Approximated Medial Axis," *Proceedings of the 2004 IEEE International Conference on Robotics & Automation*, New Orleans, LA, April 2004.

[30]Kurniawati, H. and Hsu, D., "Workspace Importance Sampling for Probabilistic Roadmap Planning," *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, September 28 - October 2, 2004.

[31]Saho, M. and Latombe, J.C., "Finding Narrow Passages with Probabilistic Roadmaps: The Small Step Retraction Method," *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2-6 August 2005.

[32]Sun, Z., Hsu, D., Jiang, T., Kurniawati, H., and Reif, J.H., "Narrow Passage Sampling for Probabilistic Roadmap Planning," *IEEE Transactions on Robotics*, Vol. 21, No. 6, December 2005.

[33]Wzorek, M. and Doherty, P., "Reconfigurable Path Planning for an Autonomous Unmanned Aerial Vehicle," paper presented at the 2006 International Conference on Hybrid Information Technology, Cheju Island, Korea, November 2006.

[34]Kim, J., Pearce, R.A., and Amato, N.M., "Extracting Optimal Paths from Roadmaps for Motion Planning," *Proceedings of the 2003 IEEE International Conference on Robotics & Automation*, Taipei, Taiwan, 14-19 September 2003.

[35]Frazzoli, E., Dahleh, M.A., and Feron, E., "Real-Time Motion Planning for Agile Autonomous Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 1, January-February 2002.

[36]Lavalle, S.M. and Hutchinson, S.A., "Optimal Motion Planning for Multiple Robots Having Independent Goals," *IEEE Transactions on Robotics and Automation*, Vol. 14, No. 6, December 1998.

[37]Cowlagi, R.V. and Tsiotras, P., "Beyond Quadtrees: Cell Decompositions for Path Planning using Wavelet Transforms," *Proceedings of the 46th IEEE Conference on Decision and Control*, New Orleans, LA, 12-14 December 2007.

[38]Zhu, D. and Latombe, J.C., "New Heurisic Algorithms for Efficient Hierarchical Path planning," *IEEE Transactions on Robotics and Automation*, Vol. 7, No. 1, February 1991.

[39]Conte, G. and Zulli, R., "Hierarchical Path Planning in a Multi-Robot Environment with a Simple Navigation Function," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 25, No. 4, April 1995.

[40]Huh, D.J., Park, J.H., Huh, U.Y., and Kim, H.I., "Path Planning and Navigation for Autonomous Mobile Robot," *IEEE 2002 28th Annual Conference of the Industrial Electronics Society*, Vol. 2, pp. 1538-1542, 5-8 November 2002.

[41]Van Den Berg, J.P. and Overmars, M.H., "Using workspace information as a guide to non-uniform sampling in probabilistic roadmap planners," *Proceedings of the 2004 IEEE International Conference on Robotics & Automation*, New Orleans, LA, April 2004.

[42]Zhang, L., Kim, Y.J., and Manocha, D., "A Hybrid Approach for Complete Motion Planning," *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, October 29 - November 2, 2007.

[43]Rosell, J., Vazquez, C., and Perez, A., "C-space decomposition using deterministic sampling and distances," *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, October 29 - November 2, 2007.

[44]Barbehenn, M. and Hutchinson, S., "Toward an Exact Incremental Geometric Robot Motion Planner," *Proceedings of the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Pittsburgh, PA, 5-9 August 1995.

[45]Lingelbach, F., "Path Planning using Probabilistic Cell Decomposition," *Proceedings of the 2004 IEEE International Conference on Robotics & Automation*, New Orleans, LA, April 2004.

[46]Lingelbach, F., "Path Planning for Mobile Manipulation using Probabilistic Cell Decomposition," *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, September 28 - October 2, 2004.

[47]Ross, I.M., *Lecture Notes in Control and Optimization*, Naval Postgraduate School, Monterey, CA, 2007.

[48]Lewis, L.R., *Rapid Motion Planning and Autonomous Obstacle Avoidance for Unmanned Vehicles*, Master's Thesis, Naval Postgraduate School, Monterey, California, December 2006.

[49]Sekhavat, P., Fleming, A., and Ross, I.M., "Time-Optimal Nonlinear Feedback Control for the NPSAT1 Spacecraft," *Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Monterey, CA, 24-28 July 2005.

[50]Bollino, K.P. and Ross, I.M., "Optimal Nonlinear Feedback Guidance for Reentry Vehicles," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, CO, 21-24 August 2006.

[51]Bollino, K.P., Lewis, L.R., Sekhavat, P., and Ross, I.M., "Pseudospectral Optimal Control: A Clear Road for Autonomous Intelligent Path Planning," *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, Rohnert Park, CA, 7-10 May 2007.

[52]Lewis, L.R. and Ross, I.M., "A Pseudospectral Method for Real-Time Motion Planning and Obstacle Avoidance," *AVT-SCI Joint Symposium on Platform Innovations and System Integration for Unmanned Air, Land and Sea Vehicles*, Florence, Italy, 14-17 May 2007.

[53]Bollino, K.P. and Lewis, L.R., "Optimal Path Planning and Control of Tactical Unmanned Aerial Vehicles in Urban Environments," *Proceedings of the AUVSI's Unmanned Systems North America 2007 Conference*, Washington, DC, August 2007.

[54]Lewis, L.R., Ross, I.M., and Gong, Q., "Pseudospectral Motion Planning Techniques for Autonomous Obstacle Avoidance," *Proceedings of the 46th IEEE Conference on Decision and Control*, New Orleans, LA, 12-14 December 2007.

[55]Ross, I.M., Sekhavat, P., Fleming, A., and Gong, Q., "Optimal Feedback Control: Foundations, Examples, and Experimental Results for a New Approach," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 2, March - April 2008.

[56]Ross, I.M., *A Beginner's Guide to DIDO: A MATLAB Application Package for Solving Optimal Control Problems*, Elissar Technical Report TR-711, http://www.elissar.biz, 2007.